LA-UR-01-6927
*Approved for public release;
distribution is unlimited.*

Title:

The Web as an API (January 2002)

Author(s):

J. H. Roman

Submitted to:

http://lib-www.lanl.gov/la-pubs/00796645.pdf

Los Alamos National Laboratory, an affirmative action/equal opportunity employer, is operated by the University of California for the U.S. Department of Energy under contract W-7405-ENG-36. By acceptance of this article, the publisher recognizes that the U.S. Government retains a nonexclusive, royalty-free license to publish or reproduce the published form of this contribution, or to allow others to do so, for U.S. Government purposes. Los Alamos National Laboratory requests that the publisher identify this article as work performed under the auspices of the U.S. Department of Energy. Los Alamos National Laboratory strongly supports academic freedom and a researcher's right to publish; as an institution, however, the Laboratory does not endorse the viewpoint of a publication or guarantee its technical correctness.

FORM 836 (10/96)

The Web as an API (January 2002)

J. H. Roman

Abstract--**As programmers we have worked with many Application Development Interface API development kits. They are well suited for interaction with a particular system. A vast source of information can be made accessible by using the http protocol through the web as an API. This setup has many advantages including the vast knowledge available on setting web servers and services. Also, these tools are available on most hardware and operating system combinations. In this paper I will cover the various types of systems that can be developed this way, their advantages and some drawbacks of this approach.**

Index Terms—**Application Programmer Interface, Distributed applications, Hyper Text Transfer Protocol, Web.**

I. Introduction

As the evolution of the web continues dynamic content keeps growing; one must ask the question: “Can it be used as a data source?”

The premise of this paper is that it is possible to use the web as a data source by accessing it through a set of function; pretty much like and API. APIs are usually a set of function calls that given the right parameters will return desired information. In our collaborative work we have found a lot of useful information on the web. Making this electronic data useful is a challenge. There is no standard output format and even worst the format is out of our control.
However with some work we can write programs to access this data. Therefore the whole paradigm can be viewed as and API. By making the right function calls with the right parameters we can extract the information required from a remote system over the web. The Hyper Text Transfer Protocol (HTTP) is very flexible and powerful protocol. This transmission protocol was designed to request information from a remote server and view it on a web browser. However it can do more than that.
The protocol is pretty simple. A request in imbedded in a Universal Record Locator (URL). The URL contains a machine name, a script or static page name, and optionally parameters. The parameters are part of the URL or are transmitted via “standard out” of the requesting process. The source machine is uniquely identified. The source machine in turn runs a web server process to accepts the request and returns a series of bytes to the consumer. By convention the protocol has some header information both ways to identify the type of information being sent in either direction. The protocol is simple yet it allows two computer programs to exchange information without much knowledge of the underlying system. The only thing in common is the one way communication pipe between them.

Manuscript received November 1, 2001 This work was supported in part by the U.S. Department of Energy.

J. H. Roman is with the Los Alamos National Laboratory, Los Alamos, NM 87544 USA (telephone: 505-667-5542, email: jhr@ lanl.gov).

II. Examples

A. Source

We have a web site where time crucial information is published. The site is updated and managed on a 7/24 basis. It has a built in notification system which is used to distribute crucial information as needed. However, a subset of the information contained is needed on another system which is not time crucial. So, we could have chosen to develop some elaborate mechanism to distribute the information. Instead we chose to publish the information via a query (cgi-bin script) on our web site. The data is created on demand so that it is always up to date. The remote system queries our system on a once a day basis. All we had to do to accomplish this was tell the other system what is the name of the script to call and the appropriate parameters.
To make it more user friendly we have and agreement with the consumer to notify them of changes in format. But, most importantly when we made a major overhaul to our system we kept the same look and feel. We changed the underlying package from one vendor to a totally different vendor. Our consumer was not even aware of the changes. Furthermore we do not have to disclose the internals of our system, and we don’t need to have knowledge of the system at their end.

B. Consumer

On another project we archive a daily web publication. We were able to write a simple Java program that retrieves the data files from the remote site and archives a copy of it locally. In the past to accomplish this we needed complex scripts to ftp the data back. This meant that we needed and account on their system and we had to know the file structure on their file system. Now all they are concerned is to make their daily publication at a given URL, and all we do is execute our program on a daily basis to retrieve the data.

III. Alternatives

Could we have solved the above problems in a different fashion?? Certainly, but none of them would be as light as these. Also, the fact that most of our systems are web based we had most of the pieces already in place.

Other approaches like CORBA and Java’s RMI could be more efficient alternatives but they require extensive coordination between source and consumer. Also, the type of people resources required is more specialized, plus the software often cost thousands of dollars. However if tight coupling is required this might be the right approach. If looser coupling can do the job then the web as an API approach offers many advantages.

IV. Implementation

The underlying theme across all the systems described above is the fact that the data is available through the HTTP protocol. The web as is can be used as a data source. This approach can work because data access through the internet can be easy to set up. There exists a web server for just about any hardware/operating system combination possible. Also, most Data Base (DB) vendors make it easy to publish their content on the web. It might not be pretty but, from a programming perspective it provides access to the information. The key advantage of this approach is the fact that we require little knowledge of the system providing the information and no control over it. Disadvantages are the speed and the fact that the source system might change their format without notice.

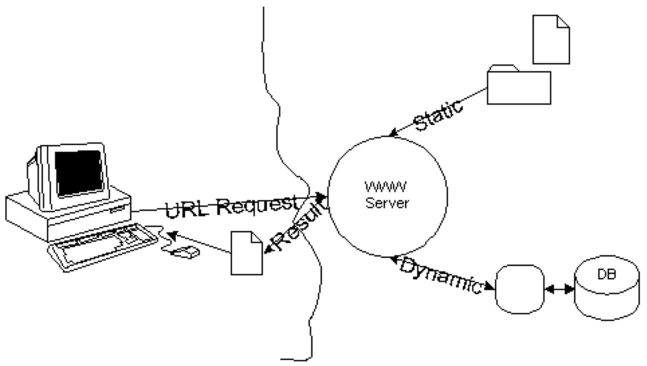

Figure1
Simple web server with dynamic content

Other advantages like the cost to retrofit an existing system or even setup a brand new machine is minimal. Web server software is usually available for free. Expertise and problem resolution answers are easy to find because the wide use of the HTTP protocol on all Web servers. The HTTP protocol is well known and understood. And what is even better is the fact that HTTP is the backbone of the internet. Because of this it is under constant review. Therefore it will continue to improve. Along this same kind of lines security can be considered as a given. Encrypted communication can be achieved easily, in particular if one uses Java as the programming language. It supports Secure Socket Layer (SSL) like most browsers. This means that current businesses accept SSL's capabilities. Therefore it is a way to provide secure transmission. The expertise to do this kind of set up is similar to setting a secure web server. There is a lot of knowledge in this area. Since web based industries rely on this kind of privacy it is certain to be part of the continuing HTTP improvements.

There are several projects that have taken the above approach and formalized. Most of them are based on eXtensible Mark-up Language (XML). XML can be used to make the whole process more robust by making it self-describing. WebDav and SOAP use it as their communication language. XML is a powerful communication language. The key here is <u>communication</u>. XML is a super set of Hyper Text Markup Language HTML. Therefore it can be said that most internet communication uses XML. HTML is a language that is good for human presentation (Business 2 People B2P type applications). However, it can be a challenge for a computer program to consume the information in this form. Not just display the tags, but to actually extract the useful information. A more structured approach like XML with a well defined tag-value scheme for the information can be read easily by a computer program. And since all the information is based on the tag-value pair the consuming program quickly identifies the tag and then can work on the value as needed. Also, to differentiate most common XML interchanges only contain pertinent information. In contrast HTML usually contains a lot of visual information.

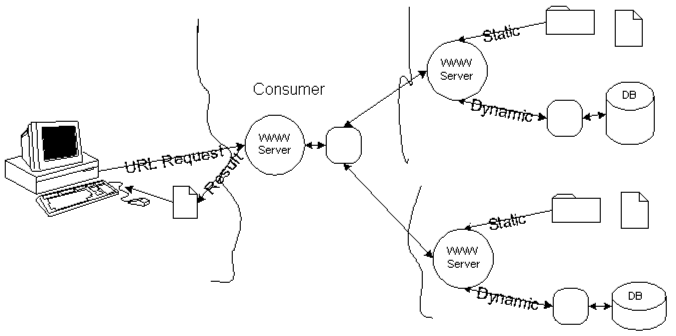

Figure 2
Sample consumer application with two sources

How would you go about setting up an application to use the basic approach?

A. Source perspective:

HTML dynamic content most often is generated by some computer program on demand. Therefore, it has some kind of repeating pattern in case of multiple results. To make it easier to access HTML tables can present the information in an organized fashion. Tables also can make it into a human readable form which is pleasant to see. However the use of a repeating pattern makes it easy to extract the information in them. Use of XML here makes is even easier to present the information in a machine readable form.

B. Consumer's perspective:

The program that consumes the data should be as flexible as possible. This way format changes at the source will not be catastrophic. The repeating pattern extraction must be built into the program for data extraction. Or even better the consuming program would read the definition from some location and use this definition to extract the data. If the definition is decoupled from the program it will be easier to change it if the source site's format changes. Again the use of XML or some other standard will greatly enhance the system and more fault tolerance can be built into it.

Most advantage from this approach can be gained when both the source and the consumer have some understanding and use a common protocol (in top of http) or at least have some agreed format. To enhance this type of system fault tolerance to network disruption must be built in so that when remote machines are not available the system degrades gracefully.

V. TYPES OF SYSTEMS

A. Distributed search engines

Some "portals" use this approach to achieve one stop searching of multiple web sites. The main search server gets the request which is translated into the appropriate request to remote distributed search engines. The results are then collected and compiled together before presented to the end user. One such system is CiriLab's Knowledge Zone (www.cirilab.com)

B. Remote Data Access

For this type of systems the data residing on a remote machine is needed by our local system to do it work. Both systems presented on the examples fit into this category. These are systems that simply consume information from a remote system and use it for its own purpose.

C. Data Exchange

Two separate systems can exchange information via this mechanism. The coupling between these systems is higher. These are the type of systems that fit into the Business 2 Business (B2B) of the Java paradigm. These systems usually mean that they both have information that the other requires. Some kind of query mechanism must be built at each end so that requests can be interchanged. The key difference here is that fact that two sets of one-way communication must be established.

D. Collaboration application

Software to allow multiple people to collectively develop a paper, work on a schedule, and other groupware type applications. One sample application is WebDAV a Distributed Authoring Protocol. This application protocol allows applications to be built that can let people author publications from distributed locations. (www.webdav.org)

E. Hierarchical Data Processing

This is a type of distributed system where the data is abstracted up at each level. Start with some raw data that is generated on a system, this data is then consumed by another system which in turn generates data to be used by another system. In this manner raw data can be shared through a set of distributed systems and it can be abstracted at each level. We built a distributed sensor architecture where each sensor publishes the information from a machine with a web server feed. The raw data is consumed and a status of the system is derived, its results are posted to the web. Then the next systems takes this data and advices on courses of action. At each level the importance of the data goes a level higher.

VI. CONCLUSION

The bottom line of this approach is that it can be a cost effective way to exchange information over the web. The expertise to implement, deploy and maintain these kind of systems is readily available. The main disadvantage could be speed. This factor is heavily dependant of the web server characteristics. Therefore this approach can be used among loosely coupled systems. For high performance tightly coupled systems other approaches could work better.

The future is only limited by our imagination. The web has been compared to the brain. Our brain makes loose associations between one piece of information and another one. And associations are built dynamically over time. HTML pages and their links resemble somewhat these relationships and the loose associations between them. So, this approach could benefit from the limitless wealth of information out there on the web. Programs that start with an HTML page and follow links given a smart algorithm can be built. Like one though leads us to another with this approach we can find better answers to the information on the web.

REFERENCES

[1] H. Bequet, "Automate SOAP calls in Java with the Proxy Patter," *JavaPro Magazine*, pp. 22-23, 86-89, Jan. 2002.

[2] Unknown, *"Brain's missing link"*, Feed Magazine: www.feedmag.com, 1998.

[3] S. StLaurent, "A look at the advantages of XML," *Chicago Tribune, www.chicago.tribune.com*, Sept 29, 1998.